

Klausur Braindump

Implementierung von Datenbanksystemen

Diverse Teilnehmer

WINTERSEMESTER 19

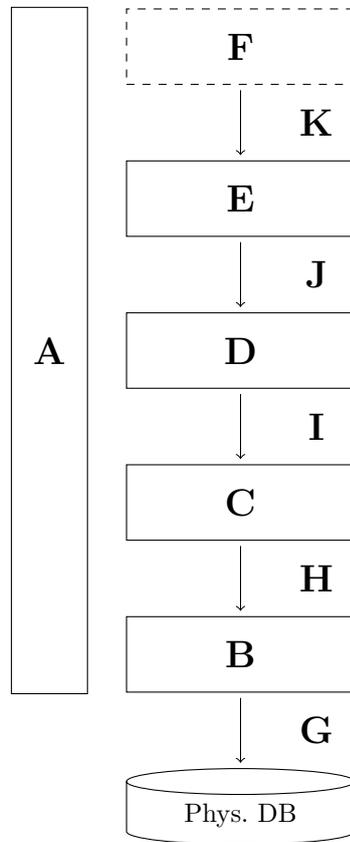
Allgemeine Anmerkungen:

- Bei dieser und allen folgenden Aufgaben – bis auf Freitext – werden *nur* die Ankreuzaufgaben gewertet. Alle Zeichnungen werden nicht beachtet.
- *Wie Immer:* Keine Garantie auf Richtigkeit. Angaben werden zum meisten Teil vereinfacht wiedergegeben. Fehler und Verbesserungen via Gitlab melden: <https://gitlab.cs.fau.de/oj14ozun/idb-ws19>.

Schichtenmodell

Teilaufgabe 1

Ordnen Sie die Platzhalter in der unteren Abbildung, den Begriffen und Beschreibungen zu:



Aufgabe 1 Wo lässt sich "Transaktion" einordnen?

- A B C D E F G H I J K

Aufgabe 2 Wo lässt sich "Blockdatei" einordnen?

- A B C D E F G H I J K

Aufgabe 3 Wo lässt sich "append(Datei, Block)" einordnen?

- A B C D E F G H I J K

Aufgabe 4 Wo lässt sich "read(TID)" einordnen?

- A B C D E F G H I J K

Aufgabe 5 Was lässt sich zum Platzhalter **F** einordnen?

- | | |
|---|--|
| <input type="checkbox"/> LRU | <input type="checkbox"/> Schattenspeicher |
| <input type="checkbox"/> Kanalkommandos | <input type="checkbox"/> <code>unfix</code> (oder <code>fix</code>) |
| <input type="checkbox"/> Kostenanalyse | <input type="checkbox"/> Overflow Buckets |
| <input type="checkbox"/> ACID | <input type="checkbox"/> SQL-String ausführen |

Aufgabe 6 Was lässt sich zum Platzhalter **J** einordnen?

- | | |
|---|--|
| <input type="checkbox"/> LRU | <input type="checkbox"/> Schattenspeicher |
| <input type="checkbox"/> Kanalkommandos | <input type="checkbox"/> <code>unfix</code> (oder <code>fix</code>) |
| <input type="checkbox"/> Kostenanalyse | <input type="checkbox"/> Overflow Buckets |
| <input type="checkbox"/> ACID | <input type="checkbox"/> SQL-String ausführen |

Aufgabe 7 Was lässt sich zum Platzhalter **D** einordnen?

- | | |
|---|--|
| <input type="checkbox"/> LRU | <input type="checkbox"/> Schattenspeicher |
| <input type="checkbox"/> Kanalkommandos | <input type="checkbox"/> <code>unfix</code> (oder <code>fix</code>) |
| <input type="checkbox"/> Kostenanalyse | <input type="checkbox"/> Overflow Buckets |
| <input type="checkbox"/> ACID | <input type="checkbox"/> SQL-String ausführen |

Aufgabe 8 Was lässt sich zum Platzhalter **B** einordnen?

- | | |
|---|--|
| <input type="checkbox"/> LRU | <input type="checkbox"/> Schattenspeicher |
| <input type="checkbox"/> Kanalkommandos | <input type="checkbox"/> <code>unfix</code> (oder <code>fix</code>) |
| <input type="checkbox"/> Kostenanalyse | <input type="checkbox"/> Overflow Buckets |
| <input type="checkbox"/> ACID | <input type="checkbox"/> SQL-String ausführen |

Aufgabe 9 Was lässt sich zum Platzhalter **G** einordnen?

- | | |
|---|--|
| <input type="checkbox"/> LRU | <input type="checkbox"/> Schattenspeicher |
| <input type="checkbox"/> Kanalkommandos | <input type="checkbox"/> <code>unfix</code> (oder <code>fix</code>) |
| <input type="checkbox"/> Kostenanalyse | <input type="checkbox"/> Overflow Buckets |
| <input type="checkbox"/> ACID | <input type="checkbox"/> SQL-String ausführen |

Hier waren noch
10-20 Fragen die-
ser Art.

Teilaufgabe 2

Aufgabe 10 Die Anwendung von ACID zum hilft bei Gerätefehlern? Ja Nein

Aufgabe 11 Die Anwendung von ACID hilft beim Mehrbenutzerbetrieb? Ja Nein

Aufgabe 12 Die Anwendung von ACID hilft die Konsistenz zu überprüfen? Ja Nein

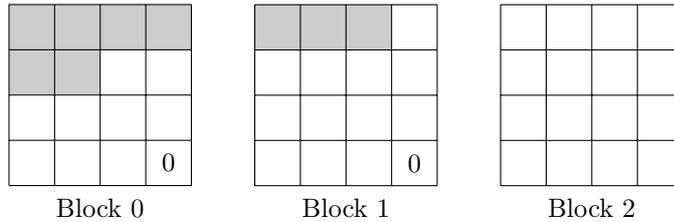
Aufgabe 13 Die Anwendung von ACID hilft beim "alles-oder-nichts" Prinzip? Ja Nein

Speicherungsstrukturen

Im Folgendem sind alle drei Segmente 16 Byte groß, wobei jeweils ein Kasten ein Byte entsprechen soll. Die Speicherung eines TID benötigt 2 Byte und ein Indexeintrag 1 Byte. Längen und Headerdaten sind bereits im Hauptspeicher, und müssen nicht behandelt werden.

Teilaufgabe 1

Bereits eingetragen sind die Sätze: TID(0,0) der Länge 6, TID(1,0) der Länge 3. **Fügen** Sie einen Satz der Länge **18** hinzu, mit dem Verfahren aus der Vorlesung. Kreuzen sie unten an, was danach gelten muss.



- Aufgabe 14** Welche Größen haben die Teilfragmente des eingefügten Satzes?
 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
- Aufgabe 15** Liegt ein Teil des eingefügten Satzes in Block 0?
 Ja Nein
- Aufgabe 16** Liegt ein Teil des eingefügten Satzes in Block 1? Ja Nein
- Aufgabe 17** Liegt ein Teil des eingefügten Satzes in Block 2? Ja Nein
- Aufgabe 18** Ist der neue Satz wird über die TID(2,0) erreichbar? Ja Nein
- Aufgabe 19** Ist der neue Satz wird über die TID(0,1) erreichbar? Ja Nein
- Aufgabe 20** Ist der neue Satz wird über die TID(1,1) erreichbar? Ja Nein
- Aufgabe 21** Ist der neu eingefügte Satz ist nicht fragmentiert? Ja Nein
- Aufgabe 22** Verwendet der neue Satz zur Fragmentierung die TID(0,1)? Ja Nein
- Aufgabe 23** Verwendet der neue Satz zur Fragmentierung die TID(1,1)? Ja Nein

Teilaufgabe 2

- Aufgabe 24** Wahlfreier Zugriff bei direkter möglich? Ja Nein
- Aufgabe 25** Ans Ende einfügen möglich bei direkter? Ja Nein
- Aufgabe 26** Ans Ende einfügen möglich bei sequentieller? Ja Nein
- Aufgabe 27** Einfügen in die Mitte möglich bei sequentieller? Ja Nein
- Aufgabe 28** Einfügen in der Mitte möglich bei direkter? Ja Nein
- Aufgabe 29** Vergrößern aller Sätze möglich bei sequentieller? Ja Nein

- Aufgabe 30** Vergrößern aller Sätze möglich bei direkter? Ja Nein
- Aufgabe 31** Löschen in der Mitte möglich bei sequentieller? Ja Nein
- Aufgabe 32** Löschen in der Mitte möglich bei direkter? Ja Nein
- Aufgabe 33** Vergrößern eines Satzes möglich bei direkter? Ja Nein
- Aufgabe 34** Vergrößern eines Satzes möglich bei sequentieller? Ja Nein

Hashing

Mittels **linearem** Hashing (Funktionsreihe: $h_j(k) = k \bmod 2^j q$, $j = 0, 1, \dots$, Initiale Bucketzahl $q = 2$) können per Bucket zwei Sätze gespeichert werden. Sobald ein Überlaufbucket benutzt wird (unabhängig vom Belegungsfaktor), um eine Satz einzufügen, muss ein *Split* durchgeführt werden.

Zum Zeitpunkt vor dem Einfügen, werden die Hashfunktionen h_0 und h_1 benutzt.

↓

0	1	2	3	4	5	6	7
5	13	6					
	11						

Es soll nun der Wert 7 eingefügt werden. Die folgenden Fragen, beziehen sich auf den Zustand, **nachdem** dieses geschehen ist:

- Aufgabe 35** Die Hashfunktion h_0 wird noch gebraucht? Ja Nein
- Aufgabe 36** Die Hashfunktion h_1 wird noch gebraucht? Ja Nein
- Aufgabe 37** Steht der Positionszeiger nach dem Einfügen auf Bucket 0? Ja Nein
- Aufgabe 38** Steht der Positionszeiger nach dem Einfügen auf Bucket 2? Ja Nein
- Aufgabe 39** Werden nach dem Einfügen 2 Overflow Buckets benutzt? Ja Nein
- Aufgabe 40** Wird nach dem Einfügen kein Overflow Buckets benutzt? Ja Nein
- Aufgabe 41** Ist der eingefügte Wert nach dem Einfügevorgang im Bucket 1? Ja Nein
- Aufgabe 42** Ist der eingefügte Wert nach dem Einfügevorgang im Bucket 3? Ja Nein
- Aufgabe 43** Wurde beim Einfügen wurde **ein** Split ausgeführt? Ja Nein
- Aufgabe 44** Wurde beim Einfügen wurde **kein** Split durchgeführt? Ja Nein
- Aufgabe 45** Befindet sich der Wert 7 Bucket 3 oder einem Überlaufbucket? Ja Nein

B*-Baum

Gegeben sei ein anfangs Leerer B*-Baum, mit $k_{\text{inner}} = 2$, $k_{\text{leaf}} = 1$. Es gilt, dass kleinere Werte ($<$) nach *links*-, und größere oder gleiche (\geq) nach *rechts* eingeordnet werden sollen.

Es sollen in der angegebenen Reihenfolge, Schlüssel und Werte eingefügt werden:

Key	4	1	6	3
Value	e	/	d	?

Beantworten Sie die folgenden Fragen, welche sich auf den Zustand des B*-Baums beziehen, *nachdem* alle Werte eingefügt wurden:

Aufgabe 46 Wie viele Unterläufe sind insgesamt eingetreten?

- 0 1 2 3 4

Aufgabe 47 Wie viele Überläufe sind insgesamt eingetreten?

- 0 1 2 3 4

Aufgabe 48 Was ist die Höhe des resultierenden Baums?

- 0 1 2 3 4

Aufgabe 49 Was ist die Anzahl der direkten Nachfolger des Wurzelknotens?

- 0 1 2 3 4

Aufgabe 50 Was ist die Anzahl der inneren Knoten (inklusive Wurzelknoten)?

- 0 1 2 3 4

Aufgabe 51 Was ist die Anzahl der Blattknoten?

- 0 1 2 3 4

Aufgabe 52 Was ist die Anzahl der Blattknoten, mit k_{leaf} Elementen?

- 0 1 2 3 4

Aufgabe 53 Wie oft kommt der Schlüssel 4 im Baum vor?

- 0 1 2 3 4

Aufgabe 54 Was ist der kleinste Wert im Wurzelknoten?

- 4 1 6 3

Pufferung

Ein Puffer mit 4 Kachelrahmen (A, B, C und D) benutzt die FLU (*Least-Frequently-Used*) Strategie, wobei in den Kontrollzuständen die Anzahl der erfolgreichen Zugriff seit Einlagerung gezählt werden.

In dieser Tabelle, soll zu drei Zeitpunkten x , y und z notiert werden, wie sich die Inhalte der Kacheln und Kontrollzustände verändern, bei den jeweils darauf folgenden Zugriffen (unterste Zeile):

Die Werte in der Tabelle sind mehr oder wenig zufällig. Die Fragen passen nicht zu den werten.

Referenzfolge		9	7	3		7	3	3	2		8	4
Zeitpunkt		x	$x + 1$	$x + 2$		y	$y + 1$	$y + 2$	$y + 3$		z	$z + 1$
A	...	7			...	4				...	2	
B	...	9			...	6				...	8	
C	...	3			...	9				...	4	
D	...	3			...	1				...	3	
E	...	2			...	1				...	10	
Kontrollzus.												
A	...	5			...	3				...	1	
B	...	5			...	10				...	2	
C	...	5			...	2				...	1	
D	...	5			...	13				...	14	
E	...	5			...	2				...	1	

Beantworten Sie aufgrund dessen die folgenden Fragen:

Aufgabe 55 In welcher Kachel befindet sich der Angefragte Wert zum Zeitpunkt $x + 1$?

- A B C D

Aufgabe 56 In welcher Kachel befindet sich der Angefragte Wert zum Zeitpunkt $x + 2$?

- A B C D

Aufgabe 57 Welcher Wert steht im verändertem Kontrollzustand *nach* $x + 1$?

- 1 2 4 7 8 9 10 14 16

Aufgabe 58 In welcher Kachel befindet sich der Angefragte Wert zum Zeitpunkt $y + 1$?

- A B C D

Aufgabe 59 In welcher Kachel befindet sich der Angefragte Wert zum Zeitpunkt $y + 2$?

- A B C D

Aufgabe 60 Welcher Wert steht im verändertem Kontrollzustand *nach* $y + 1$?

- 1 2 4 7 8 9 10 14 16

Aufgabe 61 In welcher Kachel befindet sich der Angefragte Wert zum Zeitpunkt $z + 1$?

- A B C D

Programmierschnittstelle

In einer Gaststätte, werden Reservationen mit zwei Tabellen gespeichert:

Relation **Restaurant**

Attribut	Typ
ID	Integer
Name	String
Strasse	String
Ort	String
Inhaber	String
...	...

Relation **Reservation**

Attribut	Typ
RID	Integer
Name	String
Datum	Date
Uhrzeit	String
Personenzahl	Integer
...	...

Es soll mittels JDBC, eine Ausgabe generiert werden, von allen Reservationen mit je zwei Personen, am 14. Februar 2020. Hierzu soll das folgende Gerüst vervollständigt werden:

```
import java.sql.*;

String query = "SELECT name, count(*) AS dates " +
              "FROM Restaurant, Reservierungen " +
              "WHERE datum='2020-02-14' AND " +
              "personenzahl = 2 AND " +
              "id = rid " +
              "GROUP BY name, datum";

void sucheStudentinnen() {
    try {
        BAUSTEIN-var1;
        BAUSTEIN-var2;
        BAUSTEIN-var3;
        BAUSTEIN-cond;
    } catch (SQLException se) {
        se.printStackTrace();
    }
}
```

Teilaufgabe 1

Gegeben seien zunächst die Funktionalen Bausteine, mit Lücken, gekennzeichnet durch ein "?":

```
BAUSTEIN-res: ResultSet resSet = ?.executeQuery(query);
```

```
BAUSTEIN-con: Connection con = ?.getConnection("jdbc:mysql://localhost:3306/");
```

```
BAUSTEIN-stat: Statement stat = ?.createStatement();
```

Aufgabe 62 Das ? im Baustein BAUSTEIN-res soll ersetzt werden mit:

- con stat DriverManager resSet

Aufgabe 63 Das ? im Baustein BAUSTEIN-con soll ersetzt werden mit:

- con stat DriverManager resSet

Aufgabe 64 Das ? im Baustein BAUSTEIN-stat soll ersetzt werden mit:

con stat DriverManager resSet

Teilaufgabe 2

Nun soll die Reihenfolge der Blöcke bestimmt werden:

Aufgabe 65 Im obigen Schema soll BAUSTEIN-var1 ersetzt werden durch:

BAUSTEIN-stat BAUSTEIN-con BAUSTEIN-res

Aufgabe 66 Im obigen Schema soll BAUSTEIN-var2 ersetzt werden durch:

BAUSTEIN-stat BAUSTEIN-con BAUSTEIN-res

Aufgabe 67 Im obigen Schema soll BAUSTEIN-var2 ersetzt werden durch:

BAUSTEIN-stat BAUSTEIN-con BAUSTEIN-res

Teilaufgabe 3

Analog zur Teilaufgabe 1, sind diese drei Bausteine gegeben:

BAUSTEIN-if:

```
if (?.next()) {  
    BAUSTEIN-print;  
}
```

BAUSTEIN-while:

```
while (?.next()) {  
    BAUSTEIN-print;  
}
```

BAUSTEIN-print:

```
System.out.println(?.getString(1));
```

Aufgabe 68 Das ? im Baustein BAUSTEIN-if soll ersetzt werden mit:

con stat resSet

Aufgabe 69 Das ? im Baustein BAUSTEIN-while soll ersetzt werden mit:

con stat resSet

Aufgabe 70 Das ? im Baustein BAUSTEIN-print soll ersetzt werden mit:

con stat resSet

Aufgabe 71 Im obigen Schema soll BAUSTEIN-cond ersetzt werden durch:

BAUSTEIN-if BAUSTEIN-while BAUSTEIN-print

Speichermethoden

Aufgabe 72 Müssen alle Attribute in der Projektion des C-Store aus der gleichen Relation sein?

- Ja Nein

Aufgabe 73 Darf ein Attribut in mehreren C-Store Projektionen vorkommen?

- Ja Nein

Aufgabe 74 Bei C-Store gibt es keine Aggregation-Funktion, welche aus Sortierung einen Vorteil zieht?

- Ja Nein

Aufgabe 75 Wie werden bei C-Store **wenige, unsortierte** Werte gespeichert:

- Triplets Delte-Kodierung
 Value-Pairs mit Bitmap Huffman-Kodierung

Aufgabe 76 Wie werden bei C-Store **viele, sortierte** Werte gespeichert:

- Triplets Delte-Kodierung
 Value-Pairs mit Bitmap Huffman-Kodierung

Sperren

Teilaufgabe 1

Eine Datenbank hat eine Sperrgranularität mit drei Ebenen:

Datenbank \longrightarrow Relation \longrightarrow Tupel

Beantworten sie aufgrund dessen, die folgenden Fragen:

Aufgabe 77 Ein Objekt hat von T_1 eine **SIX Sperre** erhalten. Welche Sperren sind kann T_2 dem Objekt noch geben?

X IS SIX S IX

Aufgabe 78 Ein Objekt hat von T_1 eine **S Sperre** erhalten. Welche Sperren sind kann T_2 dem Objekt noch geben?

X IS SIX S IX

Teilaufgabe 2

Drei Transaktionen T_1 , T_2 , T_3 arbeiten auf a , b und c . Zeichnen Sie den Abhängigkeitsgraphen für den folgenden Ablauf:

$r_1[a], r_1[b], c_1, w_2[a], w_4[c], r_2[c], r_3[b], r_3[b], w_2[b], c_2, r_3[b], r_3[a], c_3$

Legende: $r_X[y]$ heißt Transaktion X *liest* y . $w_X[y]$ heißt analog, dass Transaktion X , y *schreibt*. c_X bedeutet, dass Transaktion X ein **COMMIT** Befehl ausführt.

Aufgabe 79 Kreuzen Sie an, welche Kanten im Abhängigkeitsgraphen enthalten sind:

$T_1 \xrightarrow{a} T_2$

$T_1 \xrightarrow{b} T_2$

$T_1 \xrightarrow{c} T_2$

$T_1 \xrightarrow{a} T_3$

$T_1 \xrightarrow{b} T_3$

$T_1 \xrightarrow{c} T_3$

$T_2 \xrightarrow{a} T_1$

$T_2 \xrightarrow{b} T_1$

$T_2 \xrightarrow{c} T_1$

$T_2 \xrightarrow{a} T_3$

$T_2 \xrightarrow{b} T_3$

$T_2 \xrightarrow{c} T_3$

$T_3 \xrightarrow{a} T_1$

$T_3 \xrightarrow{b} T_1$

$T_3 \xrightarrow{c} T_1$

$T_3 \xrightarrow{a} T_2$

$T_3 \xrightarrow{b} T_2$

$T_3 \xrightarrow{c} T_2$

Recovery

Aufgabe 80 In welchen Methoden wird *Undo Recovery* von einem Checkpoint begrenzt?

- TOC TCC ACC

Aufgabe 81 Welche Aussagen sind wahr im Zusammenhang mit ACC (Action-Consistent-Recovery)?

- Kein Redo-Recovery notwendig
 Kein Undo-Recovery notwendig
 Redo-Recovery notwendig, aber begrenzt Checkpoint
 Undo-Recovery notwendig, aber begrenzt Checkpoint
 Redo-Recovery notwendig, und **nicht** begrenzt Checkpoint
 Undo-Recovery notwendig, und **nicht** begrenzt Checkpoint

Aufgabe 82 Welche Aussagen sind wahr im Zusammenhang mit TCC (Transaction-Consistent-Recovery)?

- Kein Redo-Recovery notwendig
 Kein Undo-Recovery notwendig
 Redo-Recovery notwendig, aber begrenzt Checkpoint
 Undo-Recovery notwendig, aber begrenzt Checkpoint
 Redo-Recovery notwendig, und **nicht** begrenzt Checkpoint
 Undo-Recovery notwendig, und **nicht** begrenzt Checkpoint

Aufgabe 83 Welche Aussagen sind wahr im Zusammenhang mit TOC (Transaction-Oriented-Checkpoints)?

- Kein Redo-Recovery notwendig
 Kein Undo-Recovery notwendig
 Redo-Recovery notwendig, aber begrenzt Checkpoint
 Undo-Recovery notwendig, aber begrenzt Checkpoint
 Redo-Recovery notwendig, und **nicht** begrenzt Checkpoint
 Undo-Recovery notwendig, und **nicht** begrenzt Checkpoint

Freitext

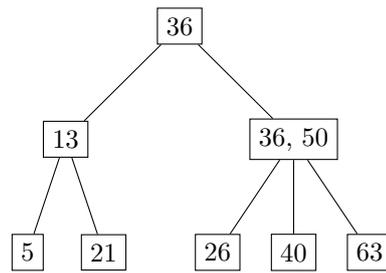
Teilaufgabe 1

Aufgabe 84 Zeichnen Sie den **nicht-Optimierten** Anfragebaum für die SQL Anfrage:

```
SELECT name, count(*) AS dates
FROM Restaurant, Reservierungen
WHERE datum='2020-02-14' AND
      personenzahl = 2 AND
      id = rid
GROUP BY name, datum
```

Teilaufgabe 2

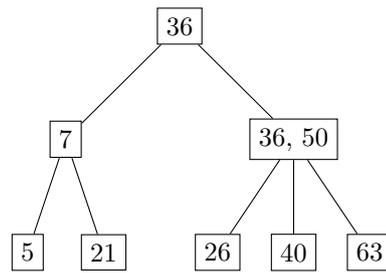
Gegeben sei der unten abgebildete B-Baum:



Aufgabe 85 Fügen Sie den Knoten **38** ein, und zeichnen Sie das Endresultat hier:

Teilaufgabe 3

Gegeben sei der unten abgebildete B-Baum:



Aufgabe 86 Löschen Sie den Knoten 7, und zeichnen Sie das Endresultat hier: